FINAL REPORT FOR THE

SYSTEM-MATIC DEVELOPMENT PROJECT

submitted by

Dr. Andrew J. McAllister

of

Faculty of Computer Science
University of New Brunswick
P.O. Box 4400
Fredericton, NB
E3B 5A3

June 27, 1990

TABLE OF CONTENTS

INTRODUCTION 3

REVIEW OF PROJECT PROPOSAL 3

OUTCOME OF THE PROJECT 4

CASE MARKET TRENDS 7

CONCLUSIONS: POTENTIAL FOR COMMERCIALIZATION 8

INTRODUCTION

This document constitutes the final report for the System-Matic Development Project (hereinafter referred to simply as "the project"). This project was funded under the Canada/New Brunswick Industrial Innovation and Technology Development Subsidiary Agreement (#9963-121-120) as per the letter of offer from the Department of Commerce and Technology dated May 10, 1988.

This final report reviews the proposed project and then describes the project results. Current market trends related to the project are discussed, along with prospects for commercializing the results of the project.

REVIEW OF PROJECT PROPOSAL

The project proposal was to design and build a Computer-Aided Software Engineering (CASE) tool called System-Matic at the University of New Brunswick in Fredericton. A CASE tool is a microcomputer-based software package used by professional software developers to help them perform various tasks involved in software development. One of the central functions of a typical CASE tool is to allow developers to create and maintain diagrams that define the software being developed. Such diagrams can be much more easily drawn, updated and checked for errors by using a microcomputer rather than performing these tasks by hand.

Commercial CASE tools have been available since 1984. At the time of the original project proposal (February 1988), approximately 1 to 2 percent of professional software developers were using CASE tools, with predictions for much larger percentages in the future. As many as 70 different vendors offered CASE tools, with the top five owning about half of the market share.

In a technical sense, most of the prominent CASE tools of 1988 supported a "Structured Systems Analysis and Design" methodology for developing software. This diagram-based method was originally designed for manual use and thus in my opinion (as well as many other industry observers) lacks the precision necessary to take full advantage of the opportunities for automation offered by CASE technology. More specifically, this imprecision means that inconsistencies can exist between:

- Two diagrams that define different subparts of a software system;
- Two diagrams that define different aspects of a given part of a system (for example, the data values used by that part and the tasks performed by that part), and;

- The system itself (that is, the computer programs) and the diagrams defining the system.

The project proposal involved defining a revised diagramming methodology based on the standard "Structured" methods but with an increased level of precision more suitable for CASE tools. Following this, a CASE tool software package called System-Matic was to be designed and constructed to support the revised diagramming methods. It was expected that the revised methods would enable System-Matic to offer the following advantages over existing tools:

- Improved consistency checking between various parts of a software system definition, including checking that descriptive diagrams are consistent with the resultant computer programs.

- An alternative "row and column" style of diagram layout that enables automatic combining, extracting, initializing and optimal rearranging of diagrams. (The term "layout" refers to arranging the components of a diagram on a page.) Unlike the standard diagram layout style, finding space to add new information to a crowded diagram is never a problem.

- Improved "reverse engineering" capabilities (that is, deriving automatically descriptive diagrams from existing computer programs).

The project work began in June 1988 and continued for 24 months, finishing on June 26, 1990. The project team consisted of Dr. Andrew J. McAllister and a full-time research assistant, Mr. Robert W. Richardson. In addition, two graduate students in Computer Science (Masters level) worked on related research topics.

OUTCOME OF THE PROJECT

The first major task to be completed was the definition of a revised diagramming methodology. This task consumed the efforts of both members of the project team from June 1988 until the end of May 1989. The resultant methodology has the following characteristics:

- The revised diagram types are similar to the industry standard "Structured Analysis" diagram types. This is intended to promote acceptance and ease of learning for the new methods.

- The mapping of information between different parts of a software system definition (that is, between diagrams or

between a diagram and the computer program it describes) is much more rigourously defined than with existing methods. This means that CASE tools based on the new methods can perform more **(a)** consistency checking and **(b)** automatic transfer of information (including reverse engineering) between various parts of a system definition than can existing CASE tools.

- The new types of diagrams can be drawn using a standard layout style or the alternative "row and column" layout style described above.

Four types of diagrams are included as part of the new methodology. **Entity-relationship diagrams** permit definition of the data values maintained by a software system. **Data flow diagrams** enable definition of the processes performed by a system. In addition, two types of **hierarchy charts** are provided, one for entity-relationship hierarchies and the other for data flow hierarchies. The methodology is defined in detail in a separate document entitled "The System-Matic Software Modeling Method: An Introduction", which is provided as an addendum to this report.

Following completion of the methodology definition, the project team began designing a new CASE tool based on the new diagram types. This tool is designed for the IBM OS/2 operating system, using the "Presentation Manager" interface (this is IBM's new graphical, window-based user interface for OS/2). As a first design phase, we tackled the following two tasks in parallel:

- The "user interface" for the new CASE tool was defined. This involved deciding on the appearance of the computer screens presented to users of the tool, as well as what operations would be provided by the tool. The user interface for the new CASE tool consists of a set of windows designed to enable drawing and otherwise managing (deleting, renaming, checking for consistency, etc.) the new types of diagrams.

- A "data model definition" was also defined. This involved deciding, in detail, what types of data values must be maintained by the tool for storage of diagrams and other design information.

This design phase was completed by the end of July 1989.

As a matter of interest, the data model definition was defined using the new entity-relationship diagrams. This provided a valuable opportunity to use and evaluate the new methodology. A few minor revisions to the diagram types resulted and we were left with a firm opinion that the methods are usable and valuable.

At this point the focus of the project changed from designing to building. The period from August 1989 until the end of the project was consumed by programming. The following major components of the CASE tool were built:

- An editing and management facility for entity-relationship diagrams (ERD's).
- A prototype reverse engineering facility for translating existing databases along with the associated programs into ERD's automatically.
- A prototype document management facility for documentation generated as part of the software development process with CASE tools.

Of these three, the ERD facility is by far the largest component. The second and third items were completed as Masters theses.

The ERD facility is approximately 80 percent complete. To provide a relatively complete CASE environment, an editing and management facility for data flow diagrams must also be built. We believe that building this second editor will take about 25 percent of the time taken to build the ERD editor because of the similarities between the two facilities (This means that all of the technical problems involved with building such a facility are already solved and that large portions of the ERD editor can be shared with the new editor.)

In retrospect, our inability to build a more complete CASE environment during the time frame of the project can be attributed to the following factors:

- The definition of the revised diagramming methodology took slightly longer than expected. We started with the supposition that the revised diagram types should differ markedly from the standard diagram types. Instead we ended up with several relatively small but very significant differences.
- The first versions of OS/2 were just being released by IBM when the project began. Thus at the time there was very little information available about OS/2 in a technical sense. In reviewing the original proposed project timetable it is now clear that we underestimated the effort required to **learn** how to program in the OS/2 environment OS/2 programs are considerably more complex in nature than are programs for most other existing operating systems - a fact which we had no way of knowing until we were a considerable way through the project.

- Another effect of OS/2's complexity is that the effort required to **build** OS/2 programs is substantial.

The cumulative effect of the above factors is that we did not build all of the CASE tool components mentioned in the proposal. We do not feel, however, that these results reflect a poor effort. Rather, based on the above list of factors we now know that 2 man-years were insufficient to complete the entire proposed project, and also that this was not evident until very late in the project.

On the plus side, we were successful in defining a methodology, designing a CASE tool package and building a substantial portion of that tool. We feel that the remainder of the tool can be completed without further financial support.

CASE MARKET TRENDS

Today's market for CASE tools is very similar to the situation at the beginning of the project. The leading tools are still based on the standard diagramming methodologies. The same CASE vendors hold their places as the industry leaders in terms of sales and influence on the market (namely, Index Technology, Knowledgeware and Bachman Information Systems).

There have been, however, some new trends emerging in the industry that have implications for any potential commercialization of project results. The first is that IBM is (not surprisingly) emerging as a dominant influence in the CASE marketplace. They have announced a plan they call AD/Cycle as a blueprint for automated software development. AD/Cycle is very much tied in with an effort to market OS/2 as a software development platform. IBM has invested in the three companies named above in return for their commitment to develop OS/2 versions of their software. Several other CASE vendors have also announced a commitment to support AC/Cycle.

As part of AD/Cycle, IBM has announced a "CASE repository" product. The vision for this product is that all CASE tools, no matter how diverse, will store their development information in a single place in a common format. This will then enable any given tool to make use of information developed using other tools.

The implications of IBM's growing role is that (a) CASE tool vendors should be positioned to take advantage of OS/2 (which we obviously are) and (b) conformance with IBM's repository is likely to become essential within the next two years. Our repository definition is based on an entity-relationship model, which is predicted to be the basis for storing information in IBM's repository.

A second major trend in the CASE industry is increasing consolidation of market share for the leading vendors. A recent article entitled "CASE: An Industry In Flux" (CASE Outlook, Vol. 2, No. 3) described the CASE market as highly fragmented, with the top four or five vendors sharing about 50 percent of the market share. The predicted trend, however, is for the market share of the leading vendors to increase steadily as larger companies make heavier investments in CASE products. It is felt that this type of buying activity will favour the larger CASE vendors and make it more difficult for smaller vendors.

This in turn is predicted to lead to fewer CASE vendors overall. Many of the smaller and/or newer vendors are likely to fold or join with the industry leaders. The prospects for success for a small vendor attempting to compete with the established vendors are considered by many to be less than they were two years ago and are likely to diminish further in the years ahead. This seems to be due in part to the fact that CASE tools are viewed as being complex and mysterious by many potential buyers and thus these buyers are unwilling to "take a chance" with a vendor that is less well known.

CONCLUSIONS: POTENTIAL FOR COMMERCIALIZATION

The original project proposal was based on the assumption that the CASE software created by this project should be made into a complete CASE environment to compete with other such products. This assumption no longer seems valid, for the following two reasons:

- The experience of actually building a CASE tool has convinced us that it takes the resources of a fairly large firm to build and maintain a comprehensive CASE product comparable in scope to those available today. Small development teams are simply unable to keep up with the rapid increases in functionality provided by the succession of new product versions put out by the larger vendors. We have also found it to be extremely difficult to raise the large amount of capital needed for such a venture until the product is built.

- As discussed previously, the leading CASE vendors appear to be very firmly entrenched. It is becoming increasingly difficult for newcomers to wrest away market share.

We still believe very strongly, however, that the software produced by this project embodies several new concepts that are very marketable. The major difficulties arise in trying to build a huge CASE package and in attempting to compete head on with large vendors. Thus it seems that the current opportunities lie in producing niche products of modest size. Assuming that such a

niche product complements an existing product, it is very possible that a cooperative marketing venture could be arranged with the vendor of that existing product.

As described earlier, the CASE software built during this project can be completed without further financial assistance. Depending on how some of the existing products evolve over the next year or two (especially product migrations to OS/2) it is likely that our software can be made to work in concert with some existing product. Opportunities for cooperative ventures must be evaluated at that time.